This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1. (withdrawn) A computer system for database management comprising, means for storing and updating a first set of data,

indexing means for storing and updating a selected index of key values related to the first set of data, the indexing means comprising means for selectively updating the index by incrementally updating the index by incrementally adding key values to the index or by fully rebuilding the index,

heuristic determination means for selecting the incremental update of the index, or the full rebuild of the index, for a given second set of data to be added to the first set of data.

2. (withdrawn) The computer system of claim 1 in which the heuristic determination means comprises a function which takes as input index meta-data, comprising characteristics of the first set of data, the index, and the second set of data.

3. (withdrawn) The computer system of claim 2 in which the database management system is a relational database management system in which the index is stored as a binary tree and in which the index meta-data comprises estimates of the table size of the first set of data, the table size of the second set of data, and the height of the index, whereby the function is empirically defined to select the incremental update of the index or the full rebuild of the index based on the predicted relative efficiencies of the incremental update of the index and the full rebuild of the index.

B1
AT

4. (withdrawn) The computer system of claim 3 in which the function determines, for each potential binary tree height of the index, a threshold percentage of the table size of the second set of data to the table size of the first set of data, whereby the threshold percentage value for each binary tree height determines the selection of the incremental index update or the full rebuild index update.

5. (withdrawn) The computer system of claim 1 in which the heuristic determination means comprises means for the user to specify the selection of the incremental update of the index or the full rebuild of the index.

6. (withdrawn) The computer system of claim 2 further comprising a means for storing the index meta-data independent of the means for storing and updating the first set of data.

7. (withdrawn) The computer system of claim 2 in which the means for storing the index meta-data comprises a recovery history file for the first set of data.

8. (withdrawn) The computer system of claim 1 in which the selected index of key values is one of a plurality of indexes and in which the selected index is the first index on the first set of data.

9. (withdrawn) The computer system of claim 1 in which the selected index of key values is one of a plurality of indexes and in which the selected index is selected on the basis of the relative sizes of each of the plurality of indexes on the first set of data.

A1
B1

10. (withdrawn) The computer system of claim 2 further comprising means for generating values for the function of the heuristic determination means, for a specified test range of values for each of the first set of data, the index, and the second set of data.

11. (withdrawn) The computer system of claim 2, the computer system having one or more CPUs, one or more disks, a sort heap and a database bufferpool, and in which the index meta-data is defined to reflect a subset of the following characteristics: the percentage of free space in the index, the estimated size of the index after both the incremental and the rebuild updates of the index, the width of the average key value in the index, the size of the sort heap and the database bufferpool in the computer system, the number and speed of the CPUs in the computer system, and the number and speed of the disks in the system.

12. (withdrawn) A computer system for relational database management comprising,

means for storing and updating a first set of data,

indexing means for storing as a binary tree, and updating, a selected index of key values related to the first set of data the indexing means comprising means for selectively updating the index by incrementally updating the index by incrementally adding key values to the index or by fully rebuilding the index,

heuristic determination means for selecting the incremental update of the index, or the full rebuild of the index, for a given second set of data to be added to the first set of data, the heuristic determination means comprising a function which takes as input index meta-data,

the index meta-data being stored separately from the first set of data and comprising estimates of the table size of the first set of data, the table size of the second set of data, and the height of the index,

A1
B1

whereby the function is empirically defined to select the incremental update of the index or the full rebuild of the index based on the predicted relative efficiencies of the incremental update of the index and the full rebuild of the index,

the heuristic determination means furthercomprising means for the user to specify the selection of the incremental update of the index or the full rebuild of the index.

13. (withdrawn) A computer program product for use with a computer comprising a central processing unit and random access memory, said computer program product comprising a computer usable medium having computer readable code means embodied in said medium for managing a database, said computer program product comprising:

computer readable program code means for causing a computer to store and update a first set of data,

computer readable program code indexing means for causing a computer to store and update a selected index of key values related to the first set of data, the indexing means comprising means for selectively updating the index by incrementally updating the index by incrementally adding key values to the index or by fully rebuilding the index,

computer readable program code heuristic determination means for causing a computer to select the incremental update of the index, or the full rebuild of the index, for a given second set of data to be added to the first set of data.

14. (withdrawn) The computer program product of claim 13 in which the heuristic determination means comprises a function which takes as input index meta-data, comprising characteristics of the first set of data, the index, and the second set of data.

15. (withdrawn) The computer program product of claim 14 in which the database is a relational database in which the index is stored as a binary tree and in which the index meta-data

A1
B1

comprises estimates of the table size of the first set of data, the table size of the second set of data, and the height of the index, whereby the function is empirically defined to select the incremental update of the index or the full rebuild of the index based on the predicted relative efficiencies of the incremental update of the index and the full rebuild of the index.

16. (withdrawn) The computer program product of claim 15 in which the function determines, for each potential binary tree height of the index, a threshold percentage of the table size of the second set of data to the table size of the first set of data, whereby the threshold percentage value for each binary tree height determines the selection of the incremental index update or the full rebuild index update.

17. (withdrawn) The computer program product of claim 13 in which the heuristic determination means comprises means for the user to specify the selection of the incremental update of the index or the full rebuild of the index.

18. (withdrawn) A method for updating a selected index in a computer system for database management, the computer system comprising means for storing and updating a first set of data related to the index, the method comprising the steps of:

(a) heuristically selecting one of: the incremental update of the index, or the full rebuild of the index, for a given second set of data to be added to the first set of data, and

(b) selectively updating the index by incrementally updating the index by incrementally adding key values to the index or by fully rebuilding the index, as selected.

19. (withdrawn) The method of claim 18 in which the step of heuristically selecting the update of the index comprises the evaluation of a function which takes as input index meta-data, comprising characteristics of the first set of data, the index, and the second set of data.

A1
B1

20. (withdrawn) A method for updating a selected b-tree index in a computer system for relational database management, the computer system comprising means for storing and updating a first set of data related to the index, the method comprising the steps of:

(a) using a heuristic determination function to select one of: the incremental update of the index, or the full rebuild of the index, for a given second set of data to be added to the first set of data, the heuristic determination function taking as input index meta-data comprising estimates of the table size of the first set of data, the table size of the second set of data, and the height of the index, whereby the heuristic function is empirically defined to select the incremental update of the index or the full rebuild of the index based on the predicted relative efficiencies of the incremental update of the index and the full rebuild of the index, and

(b) selectively updating the index by incrementally updating the index by incrementally adding key values to the index or by fully rebuilding the index, as selected.

21. (withdrawn) The method of claim 20 in which the function comprises the step of determining, for each potential binary tree height of the index, a threshold percentage of the table size of the second set of data to the table size of the first set of data, whereby the threshold percentage value for each binary tree height determines the selection of the incremental index update or the full rebuild index update.

22. (withdrawn) The method of claim 18 further comprising the step of a user specifying the selection of the incremental update of the index, the full rebuild of the index, or the selection based on the heuristic determination function.

23. (withdrawn) The method of claim 19 further comprising the step of storing the index meta-data independent of the means for storing and updating the first set of data.

A1
B1

24. (withdrawn) A computer program product tangibly embodying a program of instructions executable by a computer to perform the method steps of claim 18.

25. (withdrawn) A computer program product tangibly embodying a program of instructions executable by a computer to perform the method steps of claim 20.

26. (withdrawn) A computer program product tangibly embodying a program of instructions executable by a computer to perform the method steps of claim 21.

27. (original) A method for updating an index on a database table when data is added to the table, comprising:

receiving data records to load into the table;

selecting one of a first operation and second operation, wherein the first operation incrementally updates the index on the table as each received data record is added to the table and the second operation rebuilds the index from the table after all the received data records have been added to the table; and

using the selected first operation or second operation to update the index with the received data.

28. (original) The method of claim 27, further comprising:

determining which of the first operation or second operation is more efficient, wherein the first or second operation determined to be more efficient is the selected operation used for updating the index with the received data.

A1
B1

29. (original) The method of claim 28, wherein determining which operation is more efficient is a function of a percentage of the received data records to add to the table and characteristics of the index.

30. (original) The method of claim 29, wherein the characteristics of the index used in determining which operation is more efficient comprise a size and complexity of the index.

31. (original) The method of claim 30, wherein the index comprises a binary tree structure, and wherein a height of the index tree is indicative of the size and complexity of the index.

32. (original) The method of claim 28, wherein determining which operation is more efficient further comprises considering at least one of a following factors: an estimated time required to extract index keys from the table; an estimated time to sort the index keys; and an estimated time to rebuild the index from the sorted keys.

33. (original) The method of claim 28, further comprising:

maintaining a list of threshold values for different index sizes; and

using the number of received data records to add to the table to determine a comparison value, wherein determining whether the first or second operation is more efficient is based on the comparison value and the threshold for the size of the index to be updated.

34. (original) The method of claim 33, wherein the comparison value comprises the number of the received data records as a percentage of all data records in the table.

35. (original) The method of claim 34, wherein the index comprises a binary tree and wherein the list of threshold values provides one threshold for each of a plurality of different height index binary trees, wherein the threshold selected for comparison with the comparison value is based on the height of the index to update.

36. (original) The method of claim 33, wherein the first operation is more efficient if the comparison value is less than the threshold value and wherein the second operation is more efficient if the comparison value is greater than the threshold value.

37. (original) A system for updating an index on a database table when data is added to the table, comprising:

a database system including the table and the index on the table;

means for receiving data records to load into the table;

means for selecting one of a first operation and second operation, wherein the first operation incrementally updates the index on the table as each received data record is added to the table and the second operation rebuilds the index from the table after all the received data records have been added to the table; and

means for using the selected first operation or second operation to update the index with the received data.

38. (original) The system of claim 37, further comprising:

means for determining which of the first operation or second operation is more efficient, wherein the first or second operation determined to be more efficient is selected to use for updating the index with the received data.

A1

B1

39. (original) The system of claim 38, wherein the means for determining which operation is more efficient is a function of a percentage of the received data records to add to the table and characteristics of the index.

40. (original) The system of claim 39, wherein the characteristics of the index used in determining which operation is more efficient comprise a size and complexity of the index.

41. (original) The system of claim 40, wherein the index comprises a binary tree structure, and wherein a height of the index tree is indicative of the size and complexity of the index.

42. (original) The system of claim 38, wherein the means for determining which operation is more efficient further comprises considering at least one of a following factors: an estimated time required to extract index keys from the table; an estimated time to sort the index keys; and an estimated time to rebuild the index from the sorted keys.

43. (original) The system of claim 38, further comprising:

means for maintaining a list of threshold values for different index sizes; and

means for using the number of received data records to add to the table to determine a comparison value, wherein determining whether the first or second operation is more efficient is based on the comparison value and the threshold for the size of the index to be updated.

44. (original) The system of claim 43, wherein the comparison value comprises the number of the received data records as a percentage of all data records in the table.

45. (original) The system of claim 44, wherein the index comprises a binary tree and wherein the list of threshold values provides one threshold for each of a plurality of different height index binary trees, wherein the threshold selected for comparison with the comparison value is based on the height of the index to update.

46. (original) The system of claim 43, wherein the first operation is more efficient if the comparison value is less than the threshold value and wherein the second operation is more efficient if the comparison value is greater than the threshold value.

47. (original) A program for updating an index on a database table when data is added to the table, wherein the program is embedded in a computer readable medium and capable of causing a computer to perform:

receiving data records to load into the table;

selecting one of a first operation and second operation, wherein the first operation incrementally updates the index on the table as each received data record is added to the table and the second operation rebuilds the index from the table after all the received data records have been added to the table; and

using the first operation or second operation determined to update the index with the received data.

48. (original) The program of claim 47, wherein the program is further capable of causing the processor to perform:

determining which of the first operation or second operation is more efficient, wherein the first or second operation determined to be more efficient is selected to use for updating the index with the received data.

49. (original) The program of claim 48, wherein determining which operation is more efficient is a function of a percentage of the received data records to add to the table and characteristics of the index.

50. (original) The program of claim 49, wherein the characteristics of the index used in determining which operation is more efficient comprise a size and complexity of the index.

51. (original) The program of claim 50, wherein the index comprises a binary tree structure, and wherein a height of the index tree is indicative of the size and complexity of the index.

52. (original) The program of claim 49, wherein determining which operation is more efficient further comprises considering at least one of a following factors: an estimated time required to extract index keys from the table; an estimated time to sort the index keys; and an estimated time to rebuild the index from the sorted keys.

53. (original) The program of claim 49, wherein the program is further capable of causing the processor to perform:

maintaining a list of threshold values for different index sizes; and

using the number of received data records to add to the table to determine a comparison value, wherein determining whether the first or second operation is more efficient is based on the comparison value and the threshold for the size of the index to be updated.

54. (original) The program of claim 53, wherein the comparison value comprises the number of the received data records as a percentage of all data records in the table.

55. (original) The program of claim 54, wherein the index comprises a binary tree and wherein the list of threshold values provides one threshold for each of a plurality of different height index binary trees, wherein the threshold selected for comparison with the comparison value is based on the height of the index to update.

56. (original) The program of claim 53, wherein the first operation is more efficient if the comparison value is less than the threshold value and wherein the second operation is more efficient if the comparison value is greater than the threshold value.